

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-166977

(43)Date of publication of application : 22.06.2001

(51)Int.Cl.

G06F 12/00
// G06F 17/30

(21)Application number : 11-353630

(71)Applicant : TOSHIBA CORP

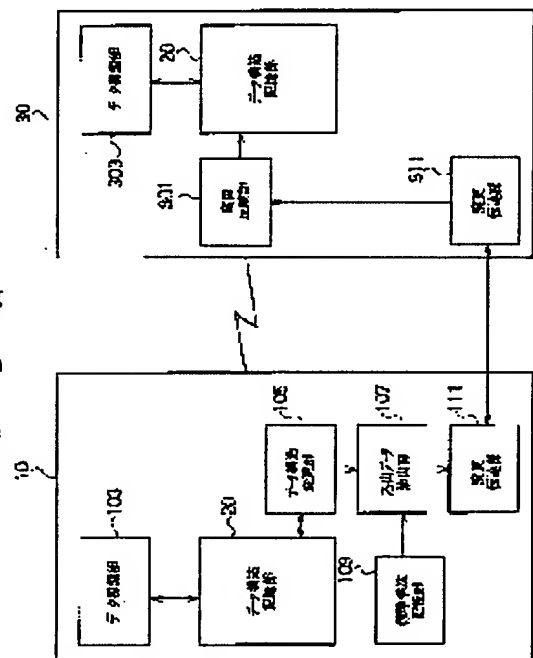
(22)Date of filing : 13.12.1999

(72)Inventor : FUJIWARA MUTSUMI

(54) DEVICE, SYSTEM AND METHOD FOR MANAGING DATA STRUCTURE AND RECORDING MEDIUM HAVING DATA STRUCTURE MANAGEMENT PROGRAM**(57)Abstract:**

PROBLEM TO BE SOLVED: To realize the simple and efficient maintenance management of a plurality of data structures arranged in one or more storage devices while reducing traffic.

SOLUTION: This data structure managing device 10 for managing a plurality of mutually equivalent data structures stored in one or more storage devices is provided with a data structure storing part 20 for storing an arbitrary first data structure, a data structure changing part 105 for changing the first data structure, a changed data extracting part 107 for extracting only the data necessary for making the first data structure at the origin of change equivalent to the second data structure at the destination of change among the changed data on the first data structure obtained by the data structure changing part 105, and a changed data transmitting part 111 for outputting the extracted changed data so that the data can be reflected on the second data structure.

**LEGAL STATUS**

[Date of request for examination]

25.09.2003

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号
特開2001-166977
(P2001-166977A)

(43) 公開日 平成13年6月22日 (2001.6.22)

(51) Int.Cl. ⁷	識別記号	F I	テーマコード(参考)
G 0 6 F 12/00	5 3 3	G 0 6 F 12/00	5 3 3 J 5 B 0 7 5
	5 2 0		5 2 0 A 5 B 0 8 2
// G 0 6 F 17/30		15/419	3 1 0

審査請求 未請求 請求項の数20 O L (全 22 頁)

(21) 出願番号 特願平11-353630

(22) 出願日 平成11年12月13日 (1999. 12. 13)

(71) 出願人 000003078

株式会社東芝

神奈川県川崎市幸区堀川町72番地

(72) 発明者 藤原 睦

神奈川県川崎市幸区柳町70番地 株式会社

東芝柳町工場内

(74) 代理人 100083806

弁理士 三好 秀和 (外7名)

Fターム(参考) 5B075 ND35 NK43 NK48 NR02 NR20

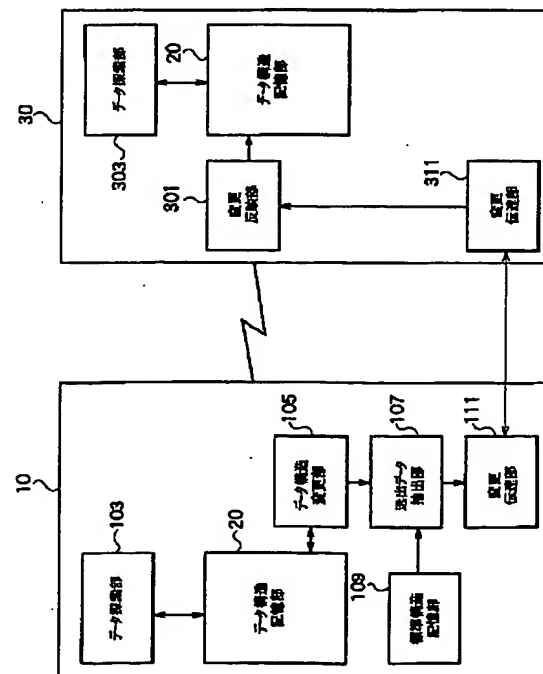
5B082 BA00 EA04 GB01 HA00 JA06

(54) 【発明の名称】 データ構造管理装置、データ構造管理システム、データ構造管理方法およびデータ構造管理プログラムを格納する記録媒体

(57) 【要約】

【課題】 1つまたは複数の記憶装置に配置された複数のデータ構造において、通信量を削減しつつ複数のデータ構造の容易かつ効率よい維持管理を実現する。

【解決手段】 1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理装置10であって、任意の第1のデータ構造を格納するデータ構造記憶部20と、第1のデータ構造を変更するデータ構造変更部105と、データ構造変更部105による第1のデータ構造上の変更データのうち、変更元の第1のデータ構造と、変更先の第2のデータ構造とを等価にするために必要なデータのみを抽出する変更データ抽出部107と、抽出された変更データを、前記第2のデータ構造に反映させるべく出力する変更データ送信部111とを具備する。



【特許請求の範囲】

【請求項1】 1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理装置であって、

任意の第1のデータ構造を格納するデータ構造記憶部と、

前記第1のデータ構造を変更するデータ構造変更部と、
前記データ構造変更部による前記第1のデータ構造上の変更データのうち、変更元の前記第1のデータ構造と、
変更先の第2のデータ構造とを等価にするために必要なデータのみを抽出する変更データ抽出部と、

前記抽出された変更データを、前記第2のデータ構造に反映させるべく出力する変更データ送信部とを具備することを特徴とするデータ構造管理装置。

【請求項2】 前記変更データ抽出部は、
前記第1のデータ構造上のそれぞれのノードが有するポインタの変更情報の一部を、前記変更データとして抽出することを特徴とする請求項1に記載のデータ構造管理装置。

【請求項3】 前記変更データ抽出部は、
ノードが削除された際に、削除されたノードのポインタの非参照状態への変更を示すデータを前記変更データとして抽出することを特徴とする請求項2に記載のデータ構造管理装置。

【請求項4】 上記データ構造管理装置は、さらに、
前記第1のデータ構造と同一のソート順序でノードを配列する標準構造を格納する標準構造記憶部を具備し、
前記変更データ抽出部は、
前記標準構造記憶部に格納される前記標準構造において、前記データ構造変更部により行われた変更と同一の変更操作により変更されるポインタの変更情報を変更データとして抽出することを特徴とする請求項2または3に記載のデータ構造管理装置。

【請求項5】 前記第1のデータ構造は、ルートノードへのポインタを格納する記憶要素を含んで構成されることを特徴とする請求項1、2、3または4に記載のデータ構造管理装置。

【請求項6】 前記変更データ抽出部は、
前記変更データを、抽出されたポインタのアドレスを含むノードの先頭からのオフセットと、前記ノードの前記第1のデータ構造上の区間の値と、前記第1のデータ構造を指示する値との組として生成することを特徴とする請求項1、2、3、4または5に記載のデータ構造管理装置。

【請求項7】 上記データ構造管理装置は、さらに、
前記第2のデータ構造上でのノードの位置を探索する位置探索部を具備し、

前記変更データ反映部は、前記変更データに基づいて、
前記位置探索部により探索された前記第2のデータ構造上のノードの位置における変更を行うことを特徴とする

請求項6に記載のデータ構造管理装置。

【請求項8】 1つまたは複数の記憶装置に格納されるデータ構造を管理するデータ構造管理装置であって、
ルートノードへのポインタを格納する記憶要素を含んで構成される任意のデータ構造を格納するデータ構造記憶部と、
前記データ構造を変更するデータ構造変更部とを具備し、

前記記憶要素は、前記記憶要素を参照する変数へのポインタを格納することを特徴とするデータ構造管理装置。

【請求項9】 1つまたは複数の記憶装置に格納されるデータ構造を管理するデータ構造管理装置であって、
ルートノードへのポインタを格納する記憶要素を含んで構成される任意のデータ構造であって、前記データ構造上のノードの少なくとも1つは、前記記憶要素へのポインタを格納するデータ構造を格納するデータ構造記憶部と、

前記データ構造を変更するデータ構造変更部とを具備することを特徴とするデータ構造管理装置。

【請求項10】 1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理装置であって、
任意の第1のデータ構造を格納するデータ構造記憶部と、

前記第1のデータ構造を変更するとともに、前記第1のデータ構造上の変更データのうち、変更元の前記第1のデータ構造と、変更先の第2のデータ構造とを等価にするために必要なデータのみを変更データとして抽出する変更データ抽出部と、

前記抽出された変更データを、前記第2のデータ構造に反映させるべく出力する変更データ送信部とを具備し、
前記変更データ抽出部は、前記第1のデータ構造上での変更を、予め定められた標準構造上での変更に変換することにより、前記変更データを算出することを特徴とするデータ構造管理装置。

【請求項11】 前記変更データ抽出部は、
前記第1のデータ構造の一部を、前記標準構造の一部に変形することにより、前記変更データを算出することを特徴とする請求項10に記載のデータ構造管理装置。

【請求項12】 前記変更データ抽出部は、
前記第1のデータ構造上で、変更対象ノードおよび前記変更対象ノードの左隣および／または右隣のノードを順次ルートノードに移動して前記第1のデータ構造を変形することにより、前記変更データを算出することを特徴とする請求項10または11に記載のデータ構造管理装置。

【請求項13】 1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理装置であって、
任意の第1のデータ構造と等価である第2のデータ構造

を格納するデータ構造記憶部と、
前記第 1 のデータ構造の変更データのうち、変更元の前記第 1 のデータ構造と、変更先の前記第 2 のデータ構造とを等価にするために必要なデータである変更データを入力する変更データ入力部と、
前記変更データを前記第 2 のデータ構造に反映する変更データ反映部とを具備することを特徴とするデータ構造管理装置。

【請求項 14】 1 つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理システムであって、
任意の第 1 のデータ構造を格納する第 1 のデータ構造記憶部と、
前記第 1 のデータ構造を変更するデータ構造変更部と、
前記データ構造変更部による前記第 1 のデータ構造上の変更データのうち、変更元の前記第 1 のデータ構造と、変更先の第 2 のデータ構造とを等価にするために必要なデータのみを抽出する変更データ抽出部と、
前記抽出された変更データを、前記第 2 のデータ構造に反映させるべく出力する変更データ送信部とを具備する第 1 のデータ構造管理装置と、
前記第 1 のデータ構造と等価である前記第 2 のデータ構造を格納する第 2 のデータ構造記憶部と、
前記第 1 のデータ構造管理装置の前記変更データ送信部から出力される前記変更データを入力する変更データ入力部と、
前記変更データを前記第 2 のデータ構造に反映する変更データ反映部とを具備する第 2 のデータ構造管理装置とを具備することを特徴とするデータ構造管理システム。

【請求項 15】 1 つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理方法であって、
任意の第 1 のデータ構造を変更するデータ構造変更ステップと、
前記データ構造変更ステップによる前記第 1 のデータ構造上の変更データのうち、変更元の前記第 1 のデータ構造と、変更先の第 2 のデータ構造とを等価にするために必要なデータのみを抽出する変更データ抽出ステップと、
前記抽出された変更データを、前記第 2 のデータ構造に反映させるべく出力する変更データ送信ステップとを含むことを特徴とするデータ構造管理方法。

【請求項 16】 前記変更データ抽出ステップは、
前記第 1 のデータ構造上のそれぞれのノードが有するポインタの変更情報の一部を、前記変更データとして抽出することを特徴とする請求項 15 に記載のデータ構造管理方法。

【請求項 17】 上記データ構造管理方法は、さらに、
前記第 1 のデータ構造と同一のソート順序でノードを配列する標準構造を格納する標準構造格納ステップを含

み、
前記変更データ抽出ステップは、
前記標準構造格納ステップにより格納される前記標準構造において、前記データ構造変更部により行われた変更と同一の変更操作により変更されるポインタの変更情報を変更データとして抽出することを特徴とする請求項 16 に記載のデータ構造管理方法。

【請求項 18】 1 つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理処理をコンピュータに実行せしめるプログラムを格納するコンピュータ読み取り可能な記録媒体であって、
任意の第 1 のデータ構造を変更するデータ構造変更処理と、
前記データ構造変更処理による前記第 1 のデータ構造上の変更データのうち、変更元の前記第 1 のデータ構造と、変更先の第 2 のデータ構造とを等価にするために必要なデータのみを抽出する変更データ抽出処理と、
前記抽出された変更データを、前記第 2 のデータ構造に反映させるべく出力する変更データ送信処理とを含むことを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項 19】 前記変更データ抽出処理は、
前記第 1 のデータ構造上のそれぞれのノードが有するポインタの変更情報の一部を、前記変更データとして抽出することを特徴とする請求項 18 に記載のコンピュータ読み取り可能な記録媒体。

【請求項 20】 上記コンピュータ読み取り可能な記録媒体は、さらに、
前記第 1 のデータ構造と同一のソート順序でノードを配列する標準構造を格納する標準構造格納処理を含み、
前記変更データ抽出処理は、
前記標準構造格納処理により格納される前記標準構造において、前記データ構造変更部により行われた変更と同一の変更操作により変更されるポインタの変更情報を変更データとして抽出することを特徴とする請求項 19 に記載のコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、データ構造管理装置、データ構造管理システム、データ構造管理方法およびデータ構造管理プログラムを格納する記録媒体に関する。特に、1 つまたは複数の計算機の記憶装置に配置された複数のデータ構造において、あるデータ構造に対する変更を、当該データ構造と他のデータ構造との間の等価性維持に必要となる変更データのみを他のデータ構造に伝達して他のデータ構造に反映することにより、通信量を削減するとともに複数のデータ構造の容易かつ効率よい維持管理を実現するための技術に関する。

【0002】

【従来の技術】従来より、計算機の主記憶を典型とする

1つまたは複数の番地付けされたランダムアクセス記憶装置（以下、「RAM」と称する）の上に、ポインタを一定の手法で格納することによって種々のデータ構造が構築され、情報表現の効率化及び操作の高速化が図られてきた。

【0003】これらデータ構造として、例えば、線形リスト、二分木、木、種々のグラフ等を、あるノードに対応づけられたRAMの一部に、他のノードへのポインタを格納することによってデータ間の構造を表現する手法がある。これらのデータ構造は一定の操作および機能を

提供するものである。

【0004】しかしながら、1つまたは複数のRAM上に配置された複数のデータ構造が等価である機能を果たすように維持および管理する手法に関しては、これまで効率的な手法が知られていなかった。

【0005】従来における複数のデータ構造の等価性維持の手法を、以下に説明する。

【0006】第1に、データ構造の機能を変更する全ての更新操作およびこの更新操作に必要な探索操作を、1つのデータ構造に対してだけでなく、複数のデータ構造のどれに対しても全く同様の内容・順序で行うことにより、これら複数のデータ構造の機能的等価性を保証する手法がある。

【0007】しかし、この第1の手法においては、更新（変更）を反映すべき他のデータ構造（以下、「コピー側データ構造」と称する）を維持および管理するのに必要な手間すなわち計算時間は、変更元のデータ構造（以下、「マスター側データ構造」と称する）を維持および管理するのに必要な手間と同一である。

【0008】具体的には、データ構造を変更するためには、多くの場合、この変更に関係する複雑な探索が実行され、この探索結果に依存して変更の内容が異なる。従って、第1の手法では、コピー側のデータ構造でもマスター側のデータ構造と同様に、必要な探索を含む変更手続きを実行しなければ双方のデータ構造の等価性を実現することができなかった。しかし、この第1の手法では、マスター側のデータ構造への変更処理とコピー側のデータ構造への変更反映処理とは、同等の計算時間（リソース）を必要とする。このため、他の処理、例えば探索のみの要求等を処理できる余地は、双方での大差がないことになる。従って、データ構造の等価性の実現に多大な計算時間を要し、例えばコピー側ではマスター側と異なった多くの探索要求を処理し、一方マスター側は変更要求の処理に専念する、等の分散環境における効率的な処理の分担が実現できなかった。すなわち、この第1の手法では、1つのデータ構造に加えられた変更のみを効率的に他のデータ構造に反映させて、他のデータ構造の管理コストを下げることは望めなかった。

【0009】第2に、ポインタの値も含めてデータ構造をRAMの内容として全く同一に保つために、変更操作

において変更されたポインタをすべて伝送して変更先に逐一反映するという手法がある。この第2の手法は、それぞれのデータ構造（より正確にはそれぞれのデータ構造を構成する各ノード）を、RAM上の同一の番地とみなせる部分に格納し、1つのデータ構造への更新（すなわち、ポインタの変更情報）を各番地の記憶内容の変更として他のデータ構造を表わすRAMにコピーする。1つの記憶装置（あるいは計算機）上でも複数の記憶装置（あるいは計算機）間でも、適切なアドレス変換を行なう手段を併用すれば、複数の等価なデータ構造を同一の番地に同一表現のポインタを格納することで構築することができる。複数のデータ構造をこのように構築し、ポインタも他のデータ（コード）と同様に単純にコピーすることによって、データ構造の変更を他のデータ構造に反映することができる。

【0010】しかし、この第2の手法は、機能的には等価であるが、ポインタ構造としては異なる形態をとることを許す複数のデータ構造に対しては適用することができない。例えば、アクセス頻度の高いノードをルートの子ノードとすべく、ポインタ構造を適宜調整する自己調整型データ構造（self-adjusting data structure）と呼ばれるデータ構造がある。この自己調整型データ構造は、機能の変更を伴わない探索操作においても、データ構造の形状を、機能的に等価な他の形状に変えてしまう。この自己調整型データ構造において、格納されるポインタの値を同一にして等価性を維持するためには、探索も含めた全ての操作によって発生するポインタ（各番地の内容）の変更を全てコピーしなければならない。このコピーのためには、膨大な量のポインタ変更情報を他のデータ構造（あるいは他のデータ構造の管理手段）に伝達しなければならない、管理の手間だけでなく通信量も非常に多くなる。

【0011】あるいは、ハイ・バランス・ツリー（high-balanced-tree）のようなデータ構造の場合には、1回の変更操作で多数のポインタが変更されることがあり、このため、伝送すべき変更データの量が多くなってしまふ。また、第2の手法においては、マスター側とコピー側とでデータ構造の構成（形態）が全く一致していなければならないという制約があった。すなわち、例えば二分木のデータ構造の場合、二分探索木（バイナリー・サーチ・ツリー：binary search tree）として等価というだけでなく、対応する同じノード間の親子関係が全て一致していなければならない。（もっとも、上記のアドレス変換が適切に行なわれれば、各ノードのアドレスおよびポインタ値まで同一である必要はない。）換言すれば、この第2の手法では、マスター側とコピー側で、バイナリー・サーチ・ツリーとしては等価であるが具体的なバイナリー・ツリーの構成が異なるデータ構造を使用することができない。

【0012】このため、例えば、マスター側とコピー側

とでツリーの構成を全く別の手法で管理する場合、すなわち、例えばマスター側は上記のハイ・バランス・ツリー (high-balanced-tree) を採用してツリーの構成を管理し、一方コピー側はスプレー・ツリー (splay tree) を採用して、探索操作が行なわれた場合にもツリーの構成を変更するような場合には、マスター側で変更を加える前のデータ構造の構成 (すなわち、ツリーの形状) とコピー側でその変更データを反映する前のデータ構造の構成がそもそも異なる。したがって単純にマスター側のデータ構造を変更した際、変更されたポインタを

コピー側のデータ構造に反映することはできない。
【0013】特に、物理的に異なる計算機の主記憶上にそれぞれ配置された複数のデータ構造の等価性を、上記の第1あるいは第2の手法により実現しようとするれば、これら計算機間を接続する通信路 (ネットワーク) に多大な負荷をかけることになり、通信路を含むシステム全体に重大な影響を及ぼす。

【0014】さらに、例えば、更新操作の場合にのみデータ構造の形状を変更するデータ構造である各種の平衡木等においても、形状の変更に伴って多数のポインタが

変更される。このため、これらのポインタの変更情報を

全て伝達するためには、データ構造を格納した RAM の間を接続する通信路には、上記の自己調整型データ構造ほどではないにしろ、多大な負荷がかかることになる。
【0015】

【発明が解決しようとする課題】以上説明したように、従来のデータ構造管理手法には、1つまたは複数の RAM 上に配置された複数のデータ構造が等価である機能を果たすように維持および管理することは、多くの計算時間

を要するとともに通信量が大きく、非効率的であったという問題点を解決するためになされたものである。
【0016】そして、その目的とするところは、1つまたは複数の RAM (記憶領域) 上に配置されたデータ構造への変更を、少ない通信量および計算時間で、他のデータ構造に反映することによって、効率よく複数のデータ構造の等価性を維持、管理することのできるデータ構造管理装置、データ構造管理システム、データ構造管理方法およびデータ構造管理プログラムを格納する記録媒体を提供する点にある。

【0017】また、本発明の他の目的は、相互に異なるアドレスに配置されるデータ構造の一部分または全体の間での等価性を容易に維持、管理することにある。

【0018】

【課題を解決するための手段】上記の課題を解決するための本発明の特徴は、あるデータ構造に対するポインタの変更データのうち、所定の標準構造に基づいて、複数のデータ構造の間の機能的な等価性を維持するために必要となる変更データを抽出し、この抽出された変更データのみを変更を反映すべきデータ構造に送出する点にある。

【0019】かかる機能を実現するための、本発明の第1の特徴は、1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理装置であって、任意の第1のデータ構造を格納するデータ構造記憶部と、前記第1のデータ構造を変更するデータ構造変更部と、前記データ構造変更部による前記第1のデータ構造上の変更データのうち、変更元の前記第1のデータ構造と、変更先の第2のデータ構造とを等価にするために必要なデータのみを抽出する変更データ抽出部と、前記抽出された変更データを、前記第2のデータ構造に反映させるべく出力する変更データ送信部とを具備することを特徴とするデータ構造管理装置を提供する点にある。

【0020】本発明の第2の特徴は、前記変更データ抽出部は、前記第1のデータ構造上のそれぞれのノードが有するポインタの変更情報の一部を、前記変更データとして抽出する点にある。

【0021】本発明の第3の特徴は、前記変更データ抽出部は、ノードが削除された際に、削除されたノードのポインタの非参照状態への変更を示すデータを前記変更データとして抽出する点にある。

【0022】本発明の第4の特徴は、上記データ構造管理装置は、さらに、前記第1のデータ構造と同一のソート順序でノードを配列する標準構造を格納する標準構造記憶部を具備し、前記変更データ抽出部は、前記標準構造記憶部に格納される前記標準構造において、前記データ構造変更部により行われた変更と同一の変更操作により変更されるポインタの変更情報を変更データとして抽出する点にある。

【0023】本発明の第5の特徴は、前記第1のデータ構造は、ルートノードへのポインタを格納する記憶要素を含んで構成される点にある。

【0024】本発明の第6の特徴は、前記変更データ抽出部は、前記変更データを、抽出されたポインタのアドレスを含むノードの先頭からのオフセットと、前記ノードの前記第1のデータ構造上の区間の値と、前記第1のデータ構造を指示する値との組として生成する点にある。

【0025】本発明の第7の特徴は、上記データ構造管理装置は、さらに、前記第2のデータ構造上でのノードの位置を探索する位置探索部を具備し、前記変更データ反映部は、前記変更データに基づいて、前記位置探索部により探索された前記第2のデータ構造上のノードの位置における変更を行う点にある。

【0026】本発明の第8の特徴は、1つまたは複数の記憶装置に格納されるデータ構造を管理するデータ構造管理装置であって、ルートノードへのポインタを格納する記憶要素を含んで構成される任意のデータ構造を格納するデータ構造記憶部と、前記データ構造を変更するデータ構造変更部とを具備し、前記記憶要素は、前記記憶

要素を参照する変数へのポインタを格納することを特徴とするデータ構造管理装置を提供する点にある。

【0027】本発明の第9の特徴は、1つまたは複数の記憶装置に格納されるデータ構造を管理するデータ構造管理装置であって、ルートノードへのポインタを格納する記憶要素を含んで構成される任意のデータ構造であって、前記データ構造上のノードの少なくとも1つは、前記記憶要素へのポインタを格納するデータ構造を格納するデータ構造記憶部と、前記データ構造を変更するデータ構造変更部とを具備することを特徴とするデータ構造管理装置を提供する点にある。

【0028】本発明の第10の特徴は、1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理装置であって、任意の第1のデータ構造を格納するデータ構造記憶部と、前記第1のデータ構造を変更するとともに、前記第1のデータ構造上の変更データのうち、変更元の前記第1のデータ構造と、変更先の第2のデータ構造とを等価にするために必要なデータのみを変更データとして抽出する変更データ抽出部と、前記抽出された変更データを、前記第2のデータ構造に反映させるべく出力する変更データ送信部とを具備し、前記変更データ抽出部は、前記第1のデータ構造上での変更を、予め定められた標準構造上での変更に変換することにより、前記変更データを算出することを特徴とするデータ構造管理装置を提供する点にある。

【0029】本発明の第11の特徴は、前記変更データ抽出部は、前記第1のデータ構造の一部を、前記標準構造の一部に変形することにより、前記変更データを算出する点にある。

【0030】本発明の第12の特徴は、前記変更データ抽出部は、前記第1のデータ構造上で、変更対象ノードおよび前記変更対象ノードの左隣および／または右隣のノードを順次ルートノードに移動して前記第1のデータ構造を変形することにより、前記変更データを算出する点にある。

【0031】本発明の第13の特徴は、1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理装置であって、任意の第1のデータ構造と等価である第2のデータ構造を格納するデータ構造記憶部と、前記第1のデータ構造の変更データのうち、変更元の前記第1のデータ構造と、変更元の前記第2のデータ構造とを等価にするために必要なデータである変更データを入力する変更データ入力部と、前記変更データを前記第2のデータ構造に反映する変更データ反映部とを具備することを特徴とするデータ構造管理装置を提供する点にある。

【0032】本発明の第14の特徴は、1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理システムであって、任意

の第1のデータ構造を格納する第1のデータ構造記憶部と、前記第1のデータ構造を変更するデータ構造変更部と、前記データ構造変更部による前記第1のデータ構造上の変更データのうち、変更元の前記第1のデータ構造と、変更先の第2のデータ構造とを等価にするために必要なデータのみを抽出する変更データ抽出部と、前記抽出された変更データを、前記第2のデータ構造に反映させるべく出力する変更データ送信部とを具備する第1のデータ構造管理装置と、前記第1のデータ構造と等価である前記第2のデータ構造を格納する第2のデータ構造記憶部と、前記第1のデータ構造管理装置の前記変更データ送信部から出力される前記変更データを入力する変更データ入力部と、前記変更データを前記第2のデータ構造に反映する変更データ反映部とを具備する第2のデータ構造管理装置とを具備することを特徴とするデータ構造管理システムを提供する点にある。

【0033】本発明の第15の特徴は、1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理方法であって、任意の第1のデータ構造を変更するデータ構造変更ステップと、前記データ構造変更ステップによる前記第1のデータ構造上の変更データのうち、変更元の前記第1のデータ構造と、変更先の第2のデータ構造とを等価にするために必要なデータのみを抽出する変更データ抽出ステップと、前記抽出された変更データを、前記第2のデータ構造に反映させるべく出力する変更データ送信ステップとを含むことを特徴とするデータ構造管理方法を提供する点にある。

【0034】本発明の第16の特徴は、前記変更データ抽出ステップは、前記第1のデータ構造上のそれぞれのノードが有するポインタの変更情報の一部を、前記変更データとして抽出する点にある。

【0035】本発明の第17の特徴は、上記データ構造管理方法は、さらに、前記第1のデータ構造と同一のソート順序でノードを配列する標準構造を格納する標準構造格納ステップを含み、前記変更データ抽出ステップは、前記標準構造格納ステップにより格納される前記標準構造において、前記データ構造変更部により行われた変更と同一の変更操作により変更されるポインタの変更情報を変更データとして抽出する点にある。

【0036】本発明の第18の特徴は、1つまたは複数の記憶装置に格納され、相互に等価である複数のデータ構造を管理するデータ構造管理処理をコンピュータに実行せしめるプログラムを格納するコンピュータ読み取り可能な記録媒体であって、任意の第1のデータ構造を変更するデータ構造変更処理と、前記データ構造変更処理による前記第1のデータ構造上の変更データのうち、変更元の前記第1のデータ構造と、変更先の第2のデータ構造とを等価にするために必要なデータのみを抽出する変更データ抽出処理と、前記抽出された変更データを、

前記第 2 のデータ構造に反映させるべく出力する変更データ送信処理とを含むことを特徴とするコンピュータ読み取り可能な記録媒体を提供する点にある。

【0037】本発明の第 19 の特徴は、前記変更データ抽出処理は、前記第 1 のデータ構造上のそれぞれのノードが有するポイントの変更情報の一部を、前記変更データとして抽出する点にある。

【0038】本発明の第 20 の特徴は、上記コンピュータ読み取り可能な記録媒体は、さらに、前記第 1 のデータ構造と同一のソート順序でノードを配列する標準構造を格納する標準構造格納処理を含み、前記変更データ抽出処理は、前記標準構造格納処理により格納される前記標準構造において、前記データ構造変更部により行われた変更と同一の変更操作により変更されるポイントの変更情報を変更データとして抽出する点にある。

【0039】

【発明の実施の形態】以下、図面を参照して、本発明に係るデータ構造管理装置、データ構造管理システム、データ構造管理方法およびデータ構造管理プログラムを格納する記録媒体の実施の形態を、詳細に説明する。

【0040】第 1 の実施形態

以下、図 1 から図 20 を参照して、本発明に係るデータ構造管理装置、データ構造管理システム、データ構造管理方法およびデータ構造管理プログラムを格納する記録媒体の第 1 の実施形態を詳細に説明する。第 1 の実施形態は、バイナリー・サーチ・ツリー (binary search tree) のデータ構造を複数の RAM 上に構築し、これらデータ構造の間で効率よく機能的な等価性を維持する機能を提供する。

【0041】図 1 は、第 1 の実施形態に係るデータ構造管理システムの一般的な機能構成を示すブロック図である。図 1 に示すように、第 1 の実施形態に係るマスター側のデータ構造管理装置 10 は、データ構造記憶部 20 と、データ探索部 103 と、データ構造変更部 105 と、送出データ抽出部 107 と、標準構造記憶部 109 と、変更送信部 111 とにより構成される。第 1 の実施形態に係るコピー側のデータ構造管理装置 30 は、データ構造記憶部 20 と、データ探索部 303 と、変更反映部 301 と、変更受信部 311 とにより構成される。

尚、図 1 の構成は、データ構造管理装置 10 のデータ構造記憶部 20 をマスターとし、このマスターにおけるデータ構造の変更を、コピー側のデータ構造管理装置 30 のデータ構造記憶部 20 に反映させるマスター・スレーブ型の構成の一例である。第 1 の実施形態は、この他、例えばデータ構造管理装置 10 およびデータ構造管理装置 30 の構成を併せ持つ複数のデータ構造管理装置を相互に接続して、相互にデータ構造変更データを交換するピア・ツー・ピア型の構成とするなど、任意に構成することができる。また、図 1 の構成は、変更を反映すべきコピー側のデータ構造を 1 つとして説明しているが、通

信機能が複数の相手に同一の変更データを送信できればコピー側のデータ構造を複数としてもよい。

【0042】第 1 の実施形態においては、変更送信部 111 および変更受信部 311 により提供される通信機能により、伝送された変更データを変更反映部 301 がコピー側のデータ構造記憶部 20 のデータ構造に反映する。この変更データがコピー側データ構造 20 に反映された後には、マスター側データ構造 20 のデータ探索部 103 とコピーデータ構造のデータ探索部 303 とは全く同等に機能する。すなわち、双方のデータ構造 20 は、同じ探索要求に対して同じ結果を返すことができる。従って、双方のデータ構造の等価性が実現されることによって、マスター・コピー間でデータが共有できる。

【0043】マスター側およびコピー側のデータ構造記憶部 20 のデータ構造は、相互に機能的に等価な任意のデータ構造である。ここで、機能的に等価とは、各ノードの論理的な順序 (本実施形態の場合はソート順) を同じくすること、すなわち機能的に同値であることをいい、相互のポイント構造自体は異なってもよい。

【0044】データ探索部 103、303 は、アプリケーションからの要求に従い、データ構造記憶部 20 にそれぞれ格納されるデータ構造中のデータを探索する。

【0045】マスター側データ構造管理装置 10 中のデータ構造変更部 105 は、データ構造記憶部 20 a 中に格納されるマスター側データ構造を適宜変更し、変更されたポイント情報である変更データを、送出データ抽出部 107 に通知する。

【0046】送出データ抽出部 107 は、変更された変更データのうち、標準構造記憶部 109 に格納される標準となるデータ構造に照らして、データ構造の各ノードの論理的順番を変化させるポイントの情報のみを抽出し、この抽出されたポイント情報を変更送信部 111 に送出する。

【0047】変更送信部 111 は、抽出されたポイント情報を、マスター側データ構造管理装置 10 と同一 RAM 内、同一計算機内あるいはローカルまたはリモートに接続される他の計算機にある変更受信部 311 に送信する。

【0048】一方、コピー側のデータ構造管理装置 30 中の変更受信部 311 は、マスター側データ構造管理装置 10 の変更送信部 111 から受信したポイント情報 (変更データ) を、変更反映部 301 に送出する。

【0049】変更反映部 301 は、変更受信部 311 から受信したポイント情報 (変更データ) を、データ構造記憶部 20 に格納されるコピー側データ構造に反映する。

【0050】次に、第 1 の実施形態に係るデータ構造管理装置のハードウェア構成を説明する。本発明に係るデータ構造管理システムにおけるデータ構造管理装置 10

およびデータ構造管理装置30は、いわゆる汎用機、ワークステーション、PC、ネットワーク端末等の各種コンピュータ単体あるいは各コンピュータを相互接続したシステムに実装される。本発明の実施形態で用いる各コンピュータシステムのハードウェアは、各種処理を行うためのCPUと、プログラムメモリ・データメモリ等のメモリと、FD・CD等の外部記憶装置と、キーボード・マウス等の入力装置と、ディスプレイ・プリンタ等の出力装置とを備える。

【0051】尚、本発明に係るデータ管理の各種処理を実現するためのプログラムは、各種記録媒体に保存することができる。かかる記録媒体を、上記ハードウェアを具備するコンピュータにより読み出し、当該プログラムを、主記憶等の記憶装置に格納されるデータ構造にアクセスして実行することにより、本発明が実施される。ここで、記録媒体とは、例えば、メモリ・磁気ディスク・光ディスク等、プログラムを記録することができる装置全般を含む。

【0052】次に、共用すべき複数のデータ構造を、RAM上にバイナリー・サーチ・ツリーで構成する場合を例として、第1の実施形態に係るデータ構造管理システムの処理を説明する。

【0053】図2は、本発明の第1の実施形態に係るデータ構造管理システムが行う、データ構造の変更および他のデータ構造への変更の反映の処理の概略を示すフローチャートである。まず、データ構造変更部105は、マスター側のデータ構造記憶部20中に格納されるデータ構造を変更する(ステップS10)。送出データ抽出部107は、このデータ構造における変更されたポイント情報のうち、標準構造記憶部109に格納される標準構造上での構造を変化させるポイント情報を抽出する(ステップS20)。変更送信部111は、この抽出されたポイント情報を、コピー側データ構造管理装置30の変更受信部311に送信する(ステップS30)。コピー側データ構造管理装置30では、変更受信部311により受信された変更データ(ポイント情報)を変更反映部301に送出する。変更反映部301は、データ構造記憶部20に格納されるデータ構造にアクセスし、受信された変更データ(ポイント情報)によりコピー側データ構造を更新する。

【0054】まず、第1の実施形態が用いるデータ構造の一例として、1つのノードにつき2つのポイントを格納する表現手法を説明する。ここで、ポイントとは、他のノードの番地を意味する。

【0055】バイナリー・ツリーは、1つのノードにつき2つのポイントを使用するだけで、親から子および子から親への双方向のリンクを実現することができる。

【0056】図3は、この双方向のリンクを実現する、leftmost-child-right-sibling representationと呼ばれる2ポイント表現手法を説明する。

【0057】図4は、図3の2ポイント表現手法を各ノードをツリー状に接続した二分木(バイナリー・ツリー)で表現した様子を示す。

【0058】図3において、各ノードの2つのポイントは、状況に応じて次のようなノードをポイントする。

【0059】(1) 一方(右)のポイント

左の子があれば左の子を、左の子がなくなかつ右の子があれば右の子を指す。

【0060】子が全くなければnil(空ポイント)である。

【0061】(2) 他方(左)のポイント

自分が左の子でかつ右の兄弟(自分の親の右の子)があれば右の兄弟を、それ以外で親がある、すなわちルートノードでない場合は親を指す。

【0062】親がない、すなわちルートノードである場合はnil(空ポイント)である。

【0063】尚、図3のように適切に配置された図上で一覧すると見過ごしがちであるが、あるノードが左右の子のどちらか一方しかない場合には、子を指す(右の)ポイントが左右どちらの子を指しているのかはポイントを見ただけでは判断できない。この子の左右を区別するためには、各ノードについて1ビットの余分な情報が必要である。この1ビットの情報の内容は、各ノード自身が左右どちらの子かを示してもよく、一方、各ノードの子へのポイントが左右でどちらの子かを示してもよい。バイナリー・サーチ・ツリー(二分探索木)の場合、キーの比較を行って左右どちらの子かを決定するということが可能である。但し、同一のキーを複数のノードが持ちうるような場合にはこの方法は使えない。一般的に、バイナリー・ツリーの構造を作るためには、この1ビットの情報が必要である。

【0064】一方、兄弟または親を指すもう1つの(左の)ポイントに関しては、それが右の兄弟を指しているのか親を指しているのかは、最高でも2回ポイントをたどれば判定できる。すなわち、右のポイントが右の兄弟を指すのは、右のポイントが指すノードの右のポイントが指すノードの左のポイントが当該ノードに一致する場合に限られるのである。但し、このポイントから、そのノードがルートか否かの情報も得る必要がある。上記の例ではこれを空ポイントか否かで判定可能である。

【0065】上記のように、バイナリー・ツリーは、1ノードにつき2つのポイントと1ビットの情報を格納することによって実現できる。この構造によって、3つのポイントを使用する一般に知られる構造と同じく、親から左の子を決定する操作、親から右の子を決定する操作、及び子から親を決定する操作が、それぞれ所定時間以内で実行できる。但し、1つの操作につき2つ以上のポイント及びビットを参照するので、これらの操作の実行時間は3つのポイントを使用する場合より長い。逆に、各ノードの内容を設定したり変更したりする操作に

おいては、ポインタの数が少ない分実行時間が短縮される効果がある。

【0066】次に、第1の実施形態で使用される2ポインタ表現のノード構成およびツリーへの実装を具体的に説明する。

【0067】まず、第1の実施形態に係るデータ構造の、記憶装置(RAM)への実装手法およびノードのポインタのフォーマットを説明する。

【0068】使用される記憶装置は、番地付けされた記憶要素で構成されており、各記憶要素には任意の番地を格納できるものとする。すなわち、番地がポインタとして使用される。以下において、番地 m の要素に言及する場合は、 $[m]$ で表す。

【0069】図5に示すように、1つのノードのポインタ部分には、偶数番地の要素 $[2n]$ とそれに続く要素 $[2n+1]$ が充てられる。ノードを指示する場合には、そのポインタ部分の偶数番地を用いる。すなわち、 $[2n]$ および $[2n+1]$ が充てられたノードは、「ノード $2n$ 」ということになる。

【0070】これらのノードを使用して構成されるバイナリー・ツリーのルートへのポインタを別の要素に格納し、この別の要素も含めた全体を一つのデータ構造として扱う。このルートへのポインタを格納する別の記憶要素を、当該ツリーのポット(鉢)と呼ぶ。ルートノードには、親または右の兄弟へのポインタの代わりにポットへのポインタを格納する。図5に示すように、ポットの番地を m 、ルートノードを $2n$ とすると、 $[m]$ には $2n$ が、 $[2n]$ には m が設定される。一方、ポット m に対応するツリーが空のときは $[m]$ には0(nil)が設定される。このように、ノードの集まりであるバイナリー・ツリーに一意に対応するポットを導入し、これらを一体として扱う点は、第1の実施形態の特徴の1つである。

【0071】なお、番地0は nil ポインタとして用いられるので、 $[0]$ はノードとしては使用しない。但し、ポットとして使用することはできる。 nil ポインタには記憶要素が対応づけられていない番地、またはノードと

して使用しない番地と判定できる番地ならいずれの番地を用いてもよい。

【0072】図5に示すように、ルート以外のノード $2n$ が左の子でかつ右の兄弟 $2r$ がある場合、 $[2n]$ には $2r+1$ が設定される。それ以外の場合、すなわち $2n$ が左の子でかつ右の兄弟がないか、または $2n$ が右の子である場合には、 $[2n]$ には親ノード $2p$ の番地 $2p$ が設定される。前者の場合、 $[2n]$ の値は奇数(すなわち、LSBが1)なので、偶数(すなわち、LSBが0)である後者の場合とLSB(LeastSignificand Bit)の1ビットのみで区別することができる。この区別は必須ではないが、手続の簡素化と性能の向上には有効である。

【0073】一方、図5に示すように、各ノード $2n$ のもう1つのポインタ $[2n+1]$ には、そのノードに子がない場合は0(nil)が設定される。 $2n$ に左の子 $2q$ がある場合には、右の子の有無に関係なく $[2n+1]$ には $2q$ が設定される。 $2n$ に左の子がなく、右の子 $2r$ のみがある場合は、 $[2n+1]$ には $2r+1$ が設定される。前者の場合、 $[2n+1]$ の値は偶数(LSBが0)なので、奇数(LSBが1)である後者の場合とLSBの1ビットのみで区別することができる。

【0074】上記のポット、ルート、親ノード、子ノードの関係は、以下の6つの場合がある。

【0075】

- a. ツリーが空の場合のポットのみ(図6参照)
- b. 対応するツリーを有するポットの場合(図7参照)
- c. 子がないノードの場合(図8参照)
- d. 右の子のみを持つノードの場合(図9参照)
- e. 左の子のみを持つノードの場合(図10参照)
- f. 左右両方の子を持つノードの場合(図11参照)

上記の6つの場合に基づいて、ノード $2x$ からその親子兄弟のノードに関する接続情報は、例えば以下のようにして判定できる。尚、コメント中に、各判定条件について、 $2x$ が上記a.からf.のどの場合のどのノードに対応するかを示す。

【0076】

```
(1) ノード $2x$ の親、兄弟の判定
if
  ([ $2x$ ])= $2x$ ) then  $2x$ はroot、potは $[2x]$  /* b. ノード $2n$  */
else if
  (([ $2x$ ]&1)=1) then  $2x$ は左の子、親は $[[2x]-1]$ 、右の兄弟は $[2x]-1$ 
                        /* f. ノード $2q$  */
else if
  ((([ $2x$ ]+1)&1)=1) then  $2x$ は右の子、親は $[2x]$ 、左の兄弟はない
                        /* d. ノード $2r$  */
else
   $2x$ は右の子、親は $[2x]$ 、左の兄弟は $[[2x]+1]$ 
                        /* f. ノード $2r$  */
```

```
(2) ノード $2x$ の子の判定
if
```

17

18

```

([2x+1]=0) then 2xに子はない。      /* c. ノード2p */
else if
  (([2x+1]&1)=1) then 2xの左の子はなく、右の子は[2x+1]-1
                                     /* d. ノード2p */
else if
  ((([2x+1]]&1)=1)ならば2xの左の子は[2x+1]、右の子は[[2x+1]]-1
                                     /* f. ノード2p */
else
  2xの左の子は[2x+1]、右の子はない
                                     /* e. ノード2p */

```

さらに、上記のように構成されたバイナリー・ツリーを変更する手続きには、次のような手続きが可能である。

【0077】(1) 空のバイナリー・ツリーに対応するポット m に、ノード $2n$ をルートとして対応付ける。すなわち、ノード $2n$ をルートとするバイナリー・ツリーを対応づける。この場合、上記の a. から b. へ構成が変更される。[$2n$] にポインタ m を設定し、[m] にポインタ $2n$ を設定する。

【0078】(2) 右の子がないノード $2p$ に、ノード $2r$ を右の子として付加する。すなわち、ノード $2r$ をルートをとするバイナリー・ツリーを付加する。右の子がない状態が、上記の c. および e. のいずれであるかは、上記の判定手続きで識別できる。

【0079】c. から d. へ構成の変更の場合、[$2r$] にポインタ $2p$ を設定し、[$2p+1$] にポインタ $2r+1$ を設定する。

【0080】一方、e. から f. へ構成の変更の場合、[$2r$] にポインタ $2p$ を設定し、[[$2p+1$]] にポインタ $2r+1$ を設定する。

【0081】(3) 左の子がないノード $2p$ に、ノード $2q$ を左の子として付加する。すなわち、ノード $2q$ をルートをとするバイナリー・ツリーを付加する。左の子がない状態が上記の c. および e. のいずれであるかは上記の判定手続きで識別できる。

【0082】c. から e. へ構成の変更の場合、[$2q$] にポインタ $2p$ を設定し、[$2p+1$] にポインタ $2q$ を設定する。

【0083】一方、d. から f. へ構成の変更の場合、[$2q$] に [$2p+1$] を設定し、[$2p+1$] にポインタ $2q$ を設定する。

【0084】(4) ノード $2x$ あるいはノード $2x$ をルートとするサブツリーを、その親ノードから分離する。ノード $2x$ が d. e. f. の $2q$ 、 $2r$ のいずれに該当するかは上記の判定手続きで識別できる。

【0085】d. の $2r$ の場合、[$[2x] + 1$] に 0 (nil) を設定する。

【0086】e. の $2q$ の場合、[$[2x] + 1$] に 0 (nil) を設定する。

【0087】f. の $2r$ の場合、[[$[2x] + 1$]] に [$2x$] を設定する。

【0088】f. の $2q$ の場合、[[$[2x] - 1$] + 1] に [$2x$] を設定する。

【0089】図 1 2 は、図 3 および図 4 のバイナリー・ツリーを第 1 の実施形態の手法で扱ったデータ構造を、ポットも含めて正確に図示した図である。尚、各ノードの先頭番地は偶数番地である。

【0090】図 1 3 は、図 1 2 のデータ構造と、バイナリー・サーチ・ツリーとして等価な構成である。このデータ構造は、どのノードも左の子を持たない。すなわち、どのノードも右の子のみを持つかあるいは子を持たないノードである。このデータ構造は、ルートノード a から末端のノード j まで昇順または降順のキーでソートされている。尚、ポットおよびノードの番地は、図 1 2 のデータ構造と同一である。この図 1 3 の構成のデータ構造を標準構造と称する。この標準構造は、標準構造記憶部 109 に格納され、送出データ抽出部 107 により参照される。

【0091】次に、図 1 2 のデータ構造に基づいて、第 1 の実施形態に係るデータ構造管理の処理における手順を具体的に説明する。

【0092】第 1 の実施形態に係るデータ構造管理装置は、上記のバイナリー・サーチ・ツリーの 2 ポインタ表現による複数のデータ構造を、等価であるデータの複数の表現として管理し、これらの 1 つに加えられた変更を、他のデータ構造に逐次反映することによって、これらを相互に同一（あるいは等価）のバイナリー・サーチ・ツリーとして機能させる。

【0093】以下では、第 1 の実施形態に係るデータ構造管理システムが、例えば 2 台の計算機の RAM 上の同一の番地（アドレス）に、各ノードおよびポットを配置した 2 つのデータ構造を管理する場合について説明する。

【0094】(1) ノードの削除の場合

まず、2 つの計算機上のデータ構造が、いずれも図 1 2 と全く同じであるとする。ここで、変更操作として一方において 50 をキーとするルートノードを削除し、60 をキーとするノードを新たなルートノードとする構成を作るものとする。図 1 4 は、この変更操作後のデータ構造を示す。図 1 4 中において、図 1 2 と異なる、すなわち変更されたポインタは破線矢印で示す。削除されたノ

ードfの2つのポインタは別として、削除後のバイナリー・サーチ・ツリーにおいて、ポットも含めて4つのポインタが変更されていることがわかる。

【0095】ここで、第1の実施形態に係るデータ構造管理装置10の変更送信部111は、これら変更されたポインタ及びその格納番地のすべてを、逐一他方のデータ構造を管理するデータ構造管理装置の変更伝達部311に送信するのではなく、図13の標準構成に対して同一の変更操作を行なった場合に変更されるポインタのうち、ノードの先頭アドレスである偶数番地に格納されたポインタ情報だけを番地とともに送信する。

【0096】図15は、図14に対応する標準構成を示す。この図1の標準構成において、図13と異なっている4つのポインタ値、すなわち、g番地の値e、e+1番地の値g+1、f番地の値0、f+1番地の値0のうち、<g番地の値e>および<f番地の値0>という2つの番地および値の組、あるいはこのどちらか一方の組が、変更データとして他方の計算機のコピー側データ構造管理装置30に送信される。

【0097】この変更データを、変更受信部311を介して受信したコピー側のデータ構造管理装置の変更反映部301は、例えば<f番地の値0>のみが送られた場合、f番地が図12に示すように、バイナリー・ツリーのノードに使用されていることを前提にして、このf番地のノードをバイナリー・ツリーから削除する操作を行なう。このノードの削除の操作は、先に変更を行なったマスター側のデータ構造管理装置10と同じ手続で行なわれる。結果として、図14と同一である構成が変更反映部301によって作り出される。

【0098】一方、<g番地の値e>のみが送られた場合でも、図12において、図13の標準構成をとったときにg番地が持つべきポインタの値fは、バイナリー・ツリー中の中間順序でgの左隣のノードの番地を計算することで求めうる。この変更データにより変更反映部301は、このg番地のポインタ値がeに変更されたことを検知し、この変更データに対応して図15においてg+1に変更される前の図13の標準構成中のe+1番地の値がf+1であることを、上記のg番地の場合と同様にして求める。これにより、変更前のg番地の値がf、e番地の値がf+1であることがそれぞれ確認され、ノードfを削除する操作がマスター側のデータ構造管理装置10側で行なわれたことが判定できる。この判定に基づいて、コピー側のデータ構造管理装置30は、マスター側と同じ手続でf番地のノードを削除する。

【0099】要するに、標準構成に照らしてどのような変更操作が行なわれたかを正確に判定するに足る番地および値の組が、変更送信部111から変更受信部311に送信されればよい。例えば、f番地の内容を実際には0にしなくても、削除の操作に限っては、<f番地の値0>を送信することで、f番地のノードを削除したこと

をコピー側に伝えることができる。

【0100】尚、これらのaからjの番地がノードとして使用されていることをコピー側のデータ構造管理装置が判断するためには、記憶装置上に記憶内容の一部としてこれらの番地を保持しておいてこれを参照してもよい。あるいは、操作を行なう場合にはそれと知って操作しているのだから、変更送信部111において送出する番地および値の組に、当該番地がノードの(偶数)番地である旨を付加して、コピー側の変更受信部311に送信してもよい。

【0101】また、対象データ構造を全く保持していなかった(コピー側)データ構造記憶部20にマスター側データ構造全体をコピーする1つの方法として、マスター側データ構造管理装置10上に、変更履歴の出力を全て保存しておき、この変更履歴をコピー側データ構造管理装置30に逐次入力してデータ構造を復元してもよい。

【0102】上記のように、実際に変更されたポインタよりも少ないポインタの変更データを送るだけで、一方のデータ構造での変更を、他方のデータ構造に効率よく反映させることができる。

【0103】(2) ノードの追加の場合

次に、データ構造にノードを追加する場合の変更反映手続を説明する。

【0104】例えば、図14のデータ構造において、f番地のノードを再びp番地をポットとするバイナリー・サーチ・ツリーに追加することを考える。

【0105】図16は、このf番地のノードを追加した後のバイナリー・サーチ・ツリーのデータ構造を示す。まず50をキーとしてツリーを探索し、f番地のノードを図16に示すように右の子がないノードeの右の子として追加することができる。図16に対応する標準構成は、図12と同様に図13であるから、図13の標準構成と図15の構成との違いは、g番地の値f、e+1番地の値f+1、f番地の値e、f+1番地の値g+1の4つのポインタである。

【0106】マスター側の変更送信部111は、これらの4つのポインタのうち、<f番地の値e>および<g番地の値f>のいずれか一方をコピー側の変更受信部311に送信する。

【0107】<f番地の値e>を受信した場合、コピー側の変更反映部301は、図14の状態においてf番地の値が0であることを調べる。変更反映部301は、送信されてきたポインタ値が、図14に対応する図15の標準構成において、ツリーに属していなかったノードfをノードeの右の子として追加した図13の標準構成に対応するいずれかの構成、例えば図16の構成に対して図14の構成を変更する操作を指示するものと判定し、中間順序でノードeの右隣(直後)になるようにノードfを追加する操作を実行する。この追加操作の

手順は、例えば以下の手順で行うことができる。

【0108】a. 指定されたノードをxとし、中間順序でその直前（または直後）になるように、追加されるノードyを追加する。すなわち、

b. xに左（または右）の子がなければ、yをxの左（または右）の子として処理を終了する。xに左（または右）の子があれば、その左（または右）の子を新たにxとする。

【0109】c. xに右（または左）の子がなければ、yをxの右（または左）の子にして処理を終了する。xに右（または左）の子があれば、その右（または左）の子を新たにxとして再びc. を実行する。

【0110】d. 空のツリーを指定した場合は、追加されるノードxをルートとする。

【0111】一方、< g番地の値 f >を受信した場合は、変更反映部301は、ノード gがツリーに属しており、かつその標準構造におけるg番地の値がfでなくeであることを判定した上で、ノード fを中間順序でノード gの左隣（ソート順で直前）になるように追加する操作を実行する。この追加操作も、上記のa. からd. の機能により行うことができる。

【0112】いずれの場合も、図16に示す同一の構成を得ることができる。

【0113】上記のようにして、変更されたポインタのすべてを逐一送るのではなく、標準構造におけるポインタの変更の一部を送るだけで、一方に加えられた変更を他方に正確に反映してそれらの同一性（機能的な等価性）を維持することができる。尚、上記の具体例では、2つのデータ構造について説明したが、3つ以上の場合にも、直接変更を加えたデータ構造からそれ以外のデータ構造に同一のポインタ値変更データを送ることにより、複数のデータ構造間で同一性を保つことができる。

【0114】また、伝達されるポインタの変更データは、標準構造におけるポインタ値及びポインタ格納番地そのものでなくてもよい。変更操作に対応して、送出データ抽出部107で効率的に作成でき、かつ、コピー側の変更反映部301でも効率的に操作内容が復元できる、ポインタ変更データと等価な変更データであればどのような形式でも構わない。

【0115】尚、第1の実施形態においては、バイナリー・ツリーのルートへのポインタを格納した記憶要素をポットとし、ポットへのポインタをツリーのルートノードに格納している。この構成により、コピー側および/またはマスター側での変更対象ノードがルートであっても、ルート以外のノードの変更の場合と同様に、ノードに格納されたポインタの変更情報を伝達するだけで、コピー側の変更反映部301にマスター側データ構造における変更内容を正確にコピー側データ構造に反映させることができる。

【0116】一例として、< f番地の値 0 >を受信した

コピー側のデータ構造が図12に示すようにノードfをルートとする場合を説明する。[f]に格納されたポットへのポインタpを知ることによって、fを削除した後の新たなルートノード、例えばノードgの要素[g]にpを設定し、かつポットpの要素[p]にgを設定して、削除後の正しいデータ構造、例えば図14を作成することができる。

【0117】バイナリー・サーチ・ツリーに関しては、機能的に等価であっても、異なる多数の構造をとることができ、任意のノードをルートとする構造があり得る。マスター側とコピー側とでツリーの構造が異なる場合には、変更対象のノードが、マスター側ではルートでなくともコピー側ではルートになっているという状況があり得る。ルートノードにポットへのポインタを格納しない図3のような構成の場合には、ルートノードの変更に伴って、これを参照している（ポットに相当する）変数を指定して、新たなルートを参照するように変更する必要がある。マスター側では変更対象ノードがコピー側でルートになっているか否かを知ることにはできない。このため、どのノードを変更した場合でも、変更に伴ってコピー側でルートを参照する変数の変更の可能性が少しでもあれば、これらの変数のアドレスを変更情報に付加して送出しなければならない。

【0118】第1の実施形態のように、ルートにポットを指示するポインタを格納すれば、必要に応じてツリーの各ノードからルートを經由してポットのアドレスを得ることができるので、ポットのアドレスを指示されなくてもルートノードの変更操作を行うことができる。従って、マスター側からコピー側にポットのアドレスを伝達する必要はない。但し、空のポットにノードを追加する場合や、標準構造におけるルートノードを変更する場合には、ポットのアドレスが送信される。しかし、これらの場合でも、ポットのアドレスはノードの要素の新たなポインタ値としてノードのアドレスと同様に扱われ、特別な付加データとして送信されるのではない。

【0119】尚、第1の実施形態ではツリー（のルート）を参照する変数をポット1つとして説明したが、ツリーを参照する変数が複数ある場合にも、1つのポットを經由してツリーにアクセスする図19のような構成を採れば、上述の方法を何ら変更することなくマスター側とコピー側とでデータ構造の等価性を維持することができる。さらに、図19に示すように、ポット側に当該ポットを参照する各変数へのポインタを格納すれば、各変数がどのポットを参照しているかという状態の変更を、これらポット側のポインタの変更として伝達することによって処理できる。

【0120】第1の実施形態では、ポットへのポインタを、ルートノードの、親または右の兄弟へのポインタを格納する記憶要素に格納していたので、ポットを含めたデータ構造を構成するのに、何ら余分な記憶リソースを

要することがない。この他、こうしたコンパクトな実現方法の他に、図20に示すように、各ノードにポットへのポインタを格納する構成を採ることもできる。この場合には、ルートノードの親または右の兄弟へのポインタを格納する要素には、ポットへのポインタを格納する必要がない。

【0121】第1の実施形態は、バイナリー・ツリーのルートへのポインタを格納した要素をデータ構造に含め、少なくとも1つのノードにポットへのポインタを格納することによって、ポットの番地を指定しなくてもルートの変更操作を行うことができる。このため、マスター側からコピー側へ変更情報を伝達する際に、ノードに格納されたポインタの変更情報のみを送信するだけでよく、ツリー（のルート）を参照する変数の番地を付加する必要がない。また、このような構成においては、ポットは1つとは限らず、複数設けることもできる。

【0122】また、第1の実施形態で扱うバイナリー・ツリーに限らず、一般にポインタによって構成される動的なデータ構造は、その一部の記憶要素のアドレスを知って、その要素から探索等のアクセスが開始される。バイナリー・ツリーではルートノードがこのような要素であり、そこからバイナリー・サーチが適用される。動的に変更されるデータ構造では、このような要素（例えばルートノード）は変更されうるが、それに伴って、このような要素を参照している要素（変数）も変更する必要がある。これらの変数を変更するためには、これらの変数の番地あるいは番地を決定するための情報をマスター側からコピー側に伝達しなければならない。

【0123】第1の実施形態は、データ構造の一部として動的に番地が変わらない記憶要素（ポット）を設け、動的に番地が変更されうる要素（ルートノードあるいは各ノード）からこの番地が変わらない要素への対応をデータ構造の機能として記憶する、即ち、ルートノードあるいは各ノードにポットへのポインタを格納する。このため、変更操作のたびにデータ構造への参照を保持する変数の番地をマスター側からコピー側へ伝達する手間を省くことができる。

【0124】さらに、マスター側からコピー側への伝達の手間が減少するだけでなく、単独のデータ構造（例えば、マスター側のみ）の変更操作を行う際にも、データ構造変更部への入力としてポットの番地（データ構造への参照を保持する変数の番地）を指定する必要がない。これにより、変更操作の前提としてデータ構造への参照を保持する変数の番地を知らなければならないという制約が除去され、データ構造への変更操作の適用の自由度が著しく向上する。

【0125】また、例えば、データ構造からノードを削除する操作を行う際に、削除対象ノードがそのデータ構造の唯一のノードである場合には、そのデータ構造すなわちそのノードを参照している変数ポットを、無効なノ

ードを参照させないように変更する必要がある。このためには、ポットの番地を知る必要があるが、ノードから当該ノードが属するバイナリー・ツリーのポットを知る手段がなければ、削除対象ノードとはべつにポットを示す情報を指定しなければならない。この指定は、上記のように全ての削除操作において必要なわけではないが、

(1) 必要か否かの判定を行ったうえで指定する。

【0126】(2) 必要か否かに拘わらず指定する。

【0127】(3) 必要な状況に陥らないようにデータ構造の使用を制限する。例えば、常に一つは（ダミーであっても）ノードを有するように使用する。

【0128】等のいずれの方法をとっても、余計な時間や記憶リソースを要する。特に指定すべきポットを知り得ない状況で削除操作が行われるおそれがある場合は、このための回避策は上記の(3)しかない。

【0129】同様の状況は、ルートノードが削除される（削除すべきノードがたまたまルートノードであった）場合や、スプレイング操作等でルートノードが交替する場合にも生ずるため、上記の(1)、(2)または(3)のような対策に余計なリソースを要する機会は少ない。

【0130】第1の実施形態のように、ノードからポットを知る手段がデータ構造に備わっていれば、真に必要な場合以外はポットを指定する必要がなく、操作対象ノードのみを指定して変更操作を行うことができるだけでなく、上記(3)のようにデータ構造の使用を制限したり、余分なリソースを要することもない。

【0131】第1の実施形態によれば、以下の効果が得られる。

【0132】第1の実施形態に係るデータ構造管理システムは、任意のデータ構造の標準構造におけるポインタ値の変更に着目して、変更されたポインタ値のうち、標準構成を前提にすれば変更反映部301側で推定および復元可能なポインタ値を省略して、変更データを生成し、伝達する。これにより、データ構造を変更するプログラムに意識させることなく、変更データの通信量を減少させるとともに、変更反映処理の処理時間を削減することができる。

【0133】尚、第1の実施形態の変形例として、上記の追加の例において、コピー側のデータ構造が図17に示すように、図14のマスター側データ構造と異なる構成である場合にも、上記の変更反映処理を適用することができる。

【0134】< f 番地の値 e > または < g 番地の値 f > を受信して、上記の機能を使用してノード f の追加を行なうことによって、図16とバイナリー・サーチ・ツリーとしては等価な図18の構成に変更することができる。

【0135】このように、マスター側とコピー側が異なる構成であっても、あるいは複数のコピーがそれぞれ異

なる構成であっても、標準構造におけるポインタの変更の一部を伝送するだけで、それらの間の等価性を維持することができる。

【0136】第2の実施形態以下、図21から図23を参照して、本発明に係るデータ構造管理装置、データ構造管理システム、データ構造管理方法およびデータ構造管理プログラムの第2の実施形態を、第1の実施形態と異なる点についてのみ、詳細に説明する。

【0137】第2の実施形態は、第1の実施形態が行うコピー側データ構造へのポインタ情報であるアドレスの送出に替えて、相対的位置情報を送出する機能を提供する。具体的には、第2の実施形態は、データ構造の各ノードに、キーの他に「区間管理方法」（特許第2768921号）に開示されている部分区間長を格納し、区間の位置によるノードの探索およびノードに対応づけられた区間の算出を行なう。

【0138】図21は、本発明の第2の実施形態に係るデータ構造管理装置の機能構成を示すブロック図である。図21に示すように、第2の実施形態は、図1の第1の実施形態と比較して、マスター側データ構造管理装置10中に位置探索部113と、コピー側データ構造管理装置30中に位置探索部313とをさらに具備する点において、第1の実施形態の変形である。

【0139】位置探索部113および位置探索部313は、区間の位置によるノードの探索およびノードに対応づけられた区間の算出を行なう。

【0140】その他の構成は、図1の第1の実施形態と同様であるため、説明は省略する。

【0141】図22(a)、(b)は、第2の実施形態におけるデータ構造の一例を示す。図22(a)、(b)において、各ノードに対応づけられた区間の長さは全て1であり、各区間の端はそのノードのLCRオーダーにおける左からの順位を表わしている。

【0142】マスター側のデータ構造管理装置10の送出データ抽出部107bは、標準構造記憶部109中の標準構造において変更操作の結果変更されたポインタ値及びポインタの格納番地を、位置探索部113を介して位置情報として得る。この位置情報は、変更以前の状態におけるその番地を含むノードの先頭からその番地までのオフセットと、そのノードが所属するバイナリー・ツリー中でのそのノードの区間（順位）の値と、そのバイナリー・ツリーのポットの番地との組で表わされる。この位置情報は、コピー側のデータ構造管理装置30の変更受信部311に送出される。

【0143】コピー側のデータ構造管理装置30の変更反映部301は、変更受信部311により受信された相対位置情報である変更データに基づいて、ポットの番地からそのポットに対応するツリーのルートを決し、区間（順位）を指定して位置探索部313に指示する。位置探索部313は、その位置を含む区間（その順位）

の、コピー側データ構造におけるノードの番地（先頭番地）を決し、決定された先頭番地にオフセットを加えることにより、コピー側のデータ構造における対応する番地を算出する。

【0144】図22(a)、(b)においては、例えば番地fは「q番地のポットの順位1のノードのオフセット0の番地」と表わされる。また番地eは「p番地のポットの順位5のノードのオフセット0の番地」と表わされる。従って、変更送信部111が送出する変更データの内容としては、例えば< f番地の値 e >の替わりに、< q番地のポットの順位1のノードのオフセット0の番地に、p番地のポットの順位5のノードのオフセット0の番地が格納された > という情報を示す位置情報となる。

【0145】尚、データ構造の状態および変更操作に関して、第1の実施形態と異なる点は、ポット以外の番地はノードからのオフセットにより指定される点である。このために、ノードfはポットpのツリーに追加される前にも、ポットqのツリーに属していることとなる。変更操作としては、図23(a)、(b)に示すように、いったんノードfを削除してから、これをLCRオーダーでノードeの右隣の位置になるように追加する。

【0146】第2の実施形態によれば、以下の効果が得られる。

【0147】すなわち、相対的位置情報を変更データとして送出するので、マスター側のデータ構造とコピー側のデータ構造とで、対応するノードが必ずしも同一の番地を占める必要がない。このため、マスター側とコピー側とで記憶領域（アドレス空間）の大きさや使い方が異なっても、上記のデータ構造の等価性を実現でき、本発明の適用対象を著しく拡大することができる。

【0148】尚、バイナリー・ツリー（バイナリー・サーチ・ツリー）において、ツリー中のノードを特定する手法として、上記の区間管理方法による以外の手法を用いることができる。例えば、図23(a)、(b)に示すように、各ノードのキーが全て異なる場合のように、変更の対象となるノードと同じキーを持つノードが変更前後のツリー中に存在しなければ、当該ノードの特定のために、キーを用いてもよい。

【0149】また、上記の説明は、複数のデータ構造間でポットの番地が一致している場合に、ノードの番地が異なることを許すものであるが、逆にノードの番地が共通で、ポットの番地が異なることを許すようにもできる。但し、標準構造における変更に関しては、ポットの番地が変更データとして送られるのはルートである左端のノードが変わる場合だけである。

【0150】尚、上記の方法の拡張として、データ構造ごとに番地が異なってもよいノードおよびポットから成る部分に関しては、そのノードまたはポットを特定するデータ及びオフセットの組で共通の番地からそのデータ

構造における対応する番地を算出するに足るデータを、マスター側の変更送信部111bとコピー側の変更受信部311との間で送受するという手法が可能である。

【0151】第3の実施形態

以下、図24から図26を参照して、本発明に係るデータ構造管理装置、データ構造管理システム、データ構造管理方法およびデータ構造管理プログラムの第3の実施形態を、第1および第2の実施形態と異なる点についてのみ、詳細に説明する。

【0152】図24は、第3の実施形態に係るデータ構造管理装置の機能構成を示すブロック図である。図24に示すように、第3の実施形態は、第1および第2の実施形態と異なり、マスター側に標準構造記憶部109を持たず、図1のデータ構造変更部105と送出データ抽出部107の機能を併せ持つデータ構造変更・抽出部106を具備する点において、第1の実施形態の変形である。

【0153】第1および第2の実施形態においては、データ構造変更部105から出力されるポイント変更情報のうち、標準構造記憶部109に記憶されている標準構造における変更のみを送出データ抽出部107、107bが抽出あるいはさらに適宜変換して変更送信部111、111bに出力する。一方、第3の実施形態においては、データ構造記憶部20のデータ構造をデータ構造変更・抽出部106が変更する際に、変更対象のデータ構造上で変更対象ポイントの標準構造における値を算出し、必要な変更データのみを変更送信部111に出力する。

【0154】これにより、標準構造記憶部109に標準構造を保持するのに要する資源およびデータ構造の変更に伴ってこの標準構造を変更するための手間を省くことができる。同時に、データ構造変更部105から一旦出力された変更データを、標準構造に照らして抽出するという送出データ抽出部107の手間を軽減することができる。

【0155】第3の実施形態は、データ構造として、第1の実施形態と同じバイナリー・ツリーを使用する。データ構造変更・抽出部106は、変更操作に応じて、必要な変更データを抽出するためにデータ構造記憶部20に記憶されている変更対象のデータ構造そのものを使用する。

【0156】以下、第3の実施形態におけるデータ構造変更に伴う変更反映処理の処理手順を説明する。

【0157】(1) ノードの削除の場合

例えば、図12のデータ構造においてノードfを削除する際に、変更データとして<g番地の値e>または<f番地の値0>を得る必要がある。<f番地の値0>は特に標準構造に照らさなくても作成できるが、<g番地の値e>は図12のデータ構造からは容易には得られない。第3の実施形態におけるデータ構造変更・抽出部1

06は、self-adjusting binary treeの操作であるスプレイング(splaying)を使用して、図12のデータ構造を変更、すなわちノードfを削除するとともに、削除後の標準構造における<g番地の値e>を以下のように決定する。

【0158】(i)まず、削除対象のノードfにスプレイングを適用してfをルートノードとする。図12では既にfがルートノードになっているので、この操作が終わった状態と考えればよい。

【0159】(ii)次に、LCRオーダーでノードfの右隣のノードgおよび左隣のノードeを求め、gにスプレイングを適用して、図25に示すように、gをルートノードに移動する。

【0160】(iii)最後にfの左隣のノードeにスプレイングを適用してeをルートに移動すると、ノードe、fおよびgの間の構造は、図26に示すようになる。

【0161】(iv)この図26の状態ではノードfを削除すれば、ノードgはLCRオーダーで左隣のノードeの右の子となり、ノードg自身は左の子を持たない構造となる。この構造はノードf削除後の標準構造図である図15におけるeとgとの関係と一致しており、<g番地の値e>も同一である。

【0162】このように、ノードを削除する場合、削除対象ノード、右隣のノード、左隣のノードの順にスプレイングを行った後に削除を行えば、左隣のノードと右隣のノードとの間の構造は必ず、「右隣のノードは左の子を持たず、かつ左隣のノードの右の子になっている」という標準構造に一致する。これを利用して、変更送信部111は、データ構造記憶部20のデータから直接送信すべき変更データ(上記の例では<g番地の値e>)を抽出することができる。

【0163】(2) ノードの追加の場合

図17のデータ構造において、ノードfを追加する場合は、以下の手順を実行する。

【0164】(i)まず、ノードfを適切な位置に追加して、例えば図18のようなデータ構造に変更する。

【0165】(ii)次に、ノードfにスプレイングを適用してルートに移動し、さらにノードfの左隣のノードeを求めてノードeにスプレイングを適用する。

【0166】(iii)その結果、ノードfは左の子を持たず、かつ(ルートノード)eの右の子となって、ノードeとノードfとの関係は図13の標準構造に一致する。得られたこの構造から、変更送信部111が送信すべき<f番地の値e>を抽出することができる。

【0167】(iv)同様に、LCRオーダーでノードfの右隣のノードを求め、まずノードgにスプレイングを適用した後に、続けてノードfにスプレイングを適用する。

【0168】(v)その結果、ノードgは左の子を持たず、かつ(ルートノード)fの右の子となって、ノード

fとノードgとの関係は、図13に示す標準構造に一致する。この構造から、変更送信部111が送信すべき<g番地の値f>を抽出することができる。

【0169】以上の手順により、特に記憶された標準構造を参照しなくても、データ構造記憶部20のデータ構造を使用して標準構造におけるポインタ値を計算して変更送信部111に出力するようにデータ構造変更・抽出部106を構成することができる。

【0170】尚、第3の実施形態においては、スプレイングを適用して実際にデータ構造を変形した後に、所定のノードの番地内容を読みとることによって変更データの抽出が行われた。しかし、実際にこの手続きを全て行わなくても、操作の種類（ノードの削除、追加）ごとに、操作対象ノードとLCRオーダーとによりその左隣または右隣のノードが求められれば、最終的に抽出される変更データを作成することができる。この場合には、標準構造はデータ構造記憶部20のデータ構造に部分的にも保持されている必要はない。ここで抽出される変更データは、指定された標準構造におけるポインタの値であるため、データ構造変更・抽出部106は、標準構造を参照するのと等価な計算を行っているといえる。

【0171】本発明の1つの特徴は、変更送信部111から変更受信部311に伝達される変更データが標準構造におけるポインタ（変数の）値であることである。この変更データがいかなる方法によって抽出、計算されていてもその機能および効果に差異はない。従って、抽出、計算の手法に種々の方法を適用しても本発明の本旨を逸脱することはない。

【0172】尚、上記の実施形態では、データ構造（二分探索木）としてソートされたデータを扱う。この標準構造は、データのソート順とノードの結合順が対応するように設定されており、ソート順に影響を及ぼす変更操作のみがノードの結合順を変えるように構成されている。同一のソート順のデータを表していれば、データ構造が異なっても機能的な差異はない。従って、上記の実施形態は、ソートされたデータを扱うデータ構造に関して、機能的な差異を生ずる変更のみを標準構造におけるポインタの変更として抽出するものの一例である。

【0173】尚、本発明は上述した実施形態に限定されるものではなく、本発明の本旨を逸脱することなく、種々変更・変形を成し得ることが可能であることは言うまでもない。また、これらの変更・変形はすべて本発明の範囲内に含まれるものである。

【0174】

【発明の効果】以上説明したように、本発明によれば、以下の効果を奏する。

【0175】すなわち、本発明に係るデータ構造管理装置、データ構造管理システム、データ構造管理方法およびデータ構造管理プログラムを格納する記録媒体は、1つまたは複数の計算機上の、1つまたは複数の記憶領域

上の複数のデータ構造の等価性を、わずかな通信量の変更データを送受することによって容易に効率よく維持することを可能とする。また、それぞれ異なる番地に格納されるデータ構造であっても、これらの等価性維持を可能とする。また、ネットワーク上の複数の計算機間でポインタ構造を効率よく複製することができ、例えば高信頼性システムなど多様なシステムに適用できる。

【0176】このように、本発明を用いれば、データ構造をネットワーク上の複数の計算機に共有させ、これらデータ構造を共有する各計算機上で記憶領域の管理をそれぞれ独自に効率よく行うことができる。

【図面の簡単な説明】

【図1】本発明の第1の実施形態に係るデータ構造管理システムの機能構成を示すブロック図である。

【図2】本発明の第1の実施形態に係るデータ構造管理方法の概略の処理手順を示すフローチャートである。

【図3】2ポインタ表現のバイナリー・ツリーのポインタ構造を説明する図である。

【図4】図3のバイナリー・ツリーのツリー構造を説明する図である。

【図5】バイナリー・ツリーのノードに格納されるポインタ（番地）を説明する図である。

【図6】ツリーが空の場合のポットのノードのポインタを説明する図である。

【図7】ポットとルートとのノードのポインタ関係を説明する図である。

【図8】子がないノードのポインタを説明する図である。

【図9】右の子のみを持つノードのポインタを説明する図である。

【図10】左の子のみを持つノードのポインタを説明する図である。

【図11】左右両方の子を持つノードのポインタを説明する図である。

【図12】図3のバイナリー・ツリーを、ポットを含めて記述したデータ構造の一例を示す図である。

【図13】図12のデータ構造と等価である標準構造の一例を示す図である。

【図14】変更操作を行った後の図12のデータ構造の一例を示す図である。

【図15】図14と同様の変更操作を行った後の図13の標準構造の一例を示す図である。

【図16】図14のデータ構造に追加の変更操作を行った後のデータ構造の一例を示す図である。

【図17】マスター側のデータ構造と異なるコピー側のデータ構造の一例を示す図である。

【図18】図17のデータ構造に対するノードの追加操作で得られる、図16のデータ構造と等価であるデータ構造の一例を示す図である。

【図19】第1の実施形態の変形例に係るポットとこの

ポットを参照する変数とのポインタ構造の一例を説明する図である。

【図20】第1の実施形態の変形例に係るポットとデータ構造を構成する各ノードとのポインタ構造の一例を説明する図である。

【図21】本発明の第2の実施形態に係るデータ構造管理システムの機能構成を示すブロック図である。

【図22】第2の実施形態に係るデータ構造の一例を示す図である。

【図23】図22のデータ構造に対してノードfを追加して得られるデータ構造の一例を示す図である。

【図24】本発明の第3の実施形態に係るデータ構造管理システムの機能構成を示すブロック図である。

【図25】図12のデータ構造に対して、ルートノードfの右隣のノードgにスプレイングを適用してノードgをルートに移動した後のデータ構造を示す図である。

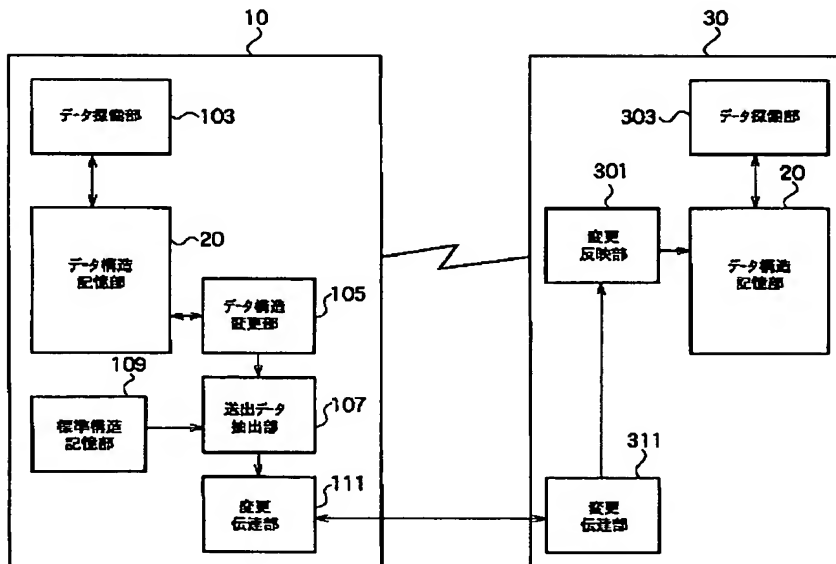
【図26】図25のデータ構造に対して、ノードfの左

隣のノードeにスプレイングを適用してノードeをルートに移動した後のデータ構造を示す図である。

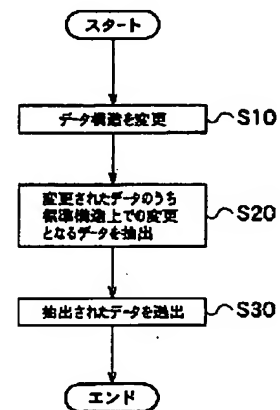
【符号の説明】

- 10 データ構造管理装置
- 20 データ構造記憶部
- 30 データ構造管理装置
- 103 データ探索部
- 105 データ構造変更部
- 106 データ構造変更・抽出部
- 107、107b 送出データ抽出部
- 109 標準構成記憶部
- 111、111b 変更送信部
- 301 変更反映部
- 303 データ探索部
- 311 変更受信部
- 313 位置探索部

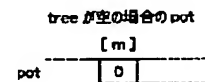
【図1】



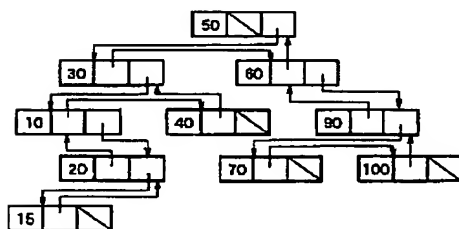
【図2】



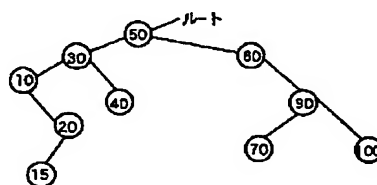
【図6】



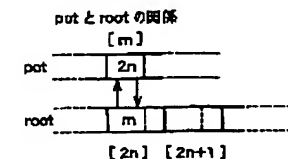
【図3】



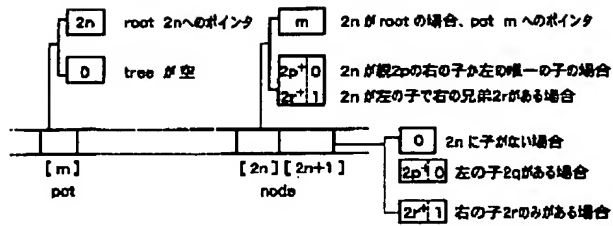
【図4】



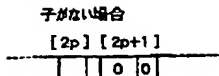
【図7】



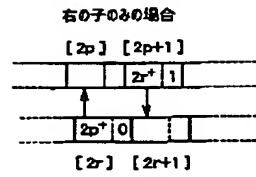
【図5】



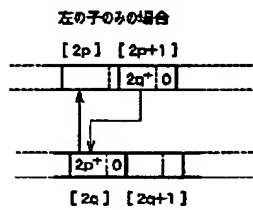
【図8】



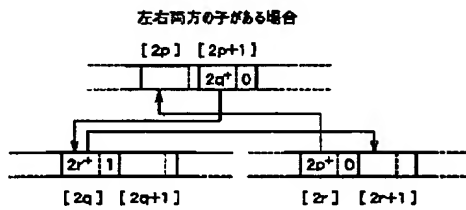
【図9】



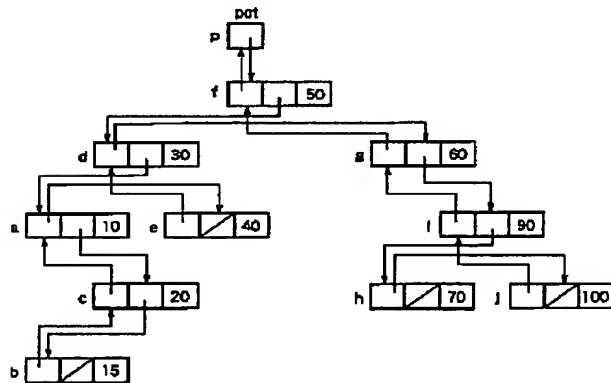
【図10】



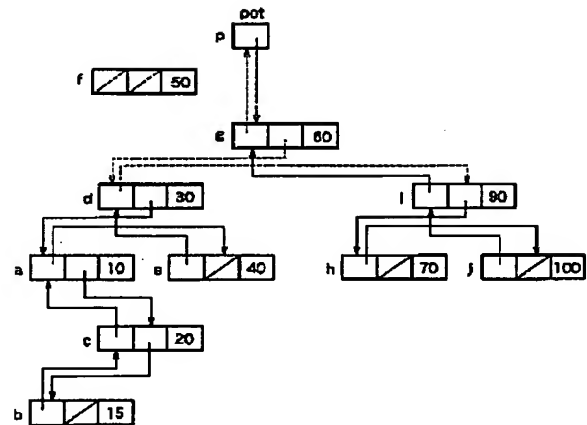
【図11】



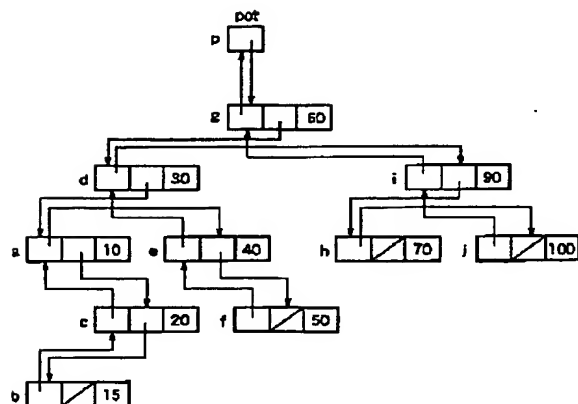
【図12】



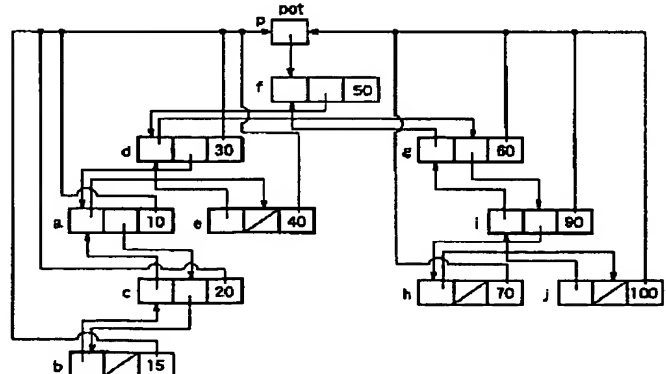
【図14】



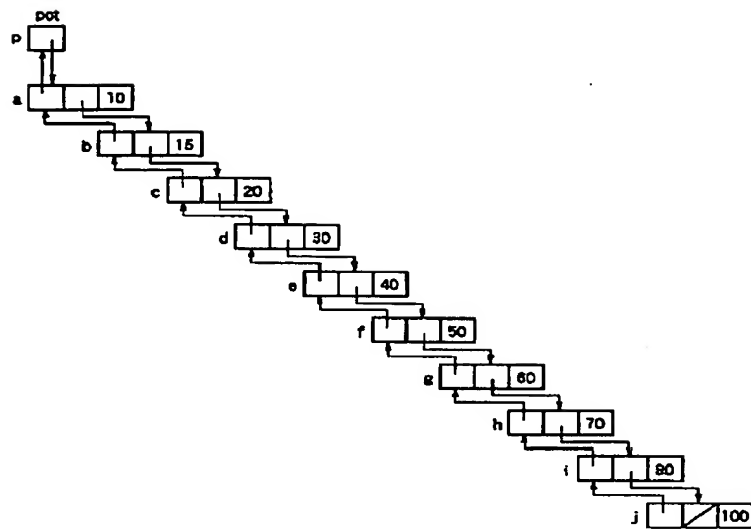
【図16】



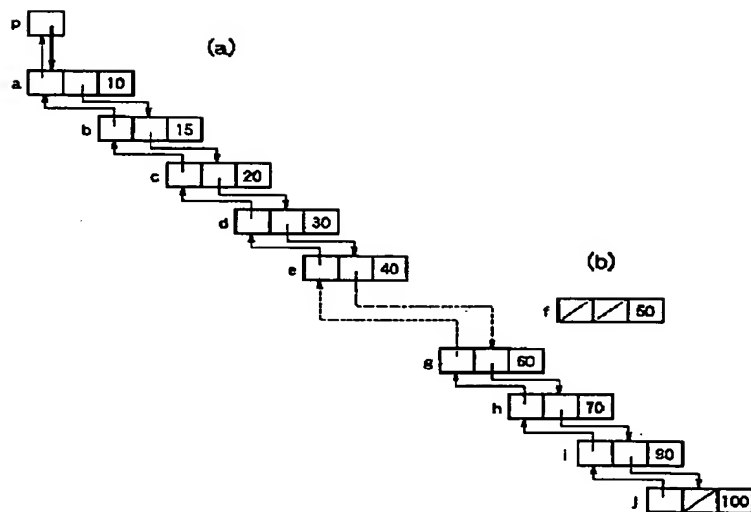
【図20】



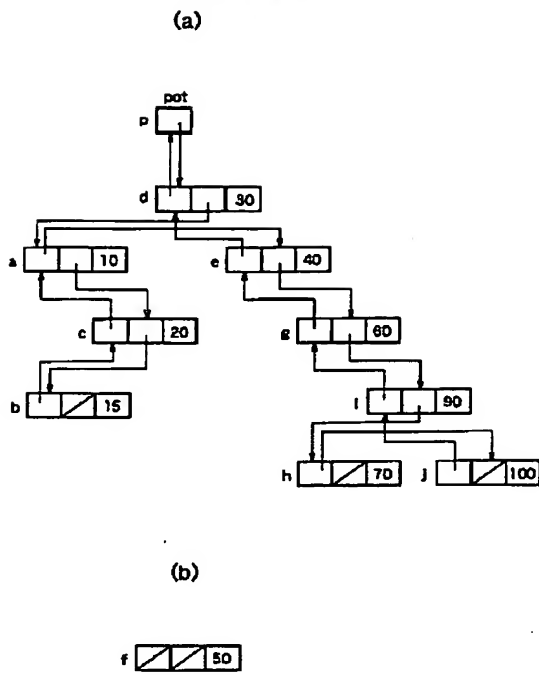
【図13】



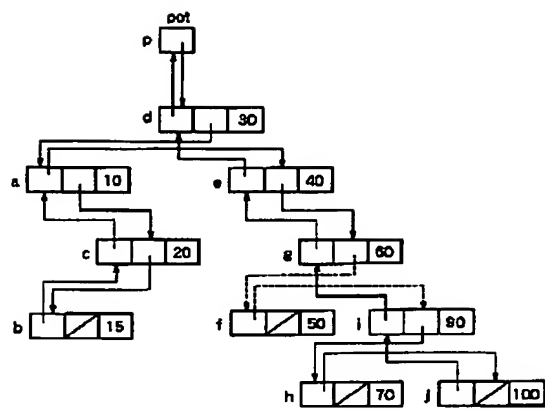
【図15】



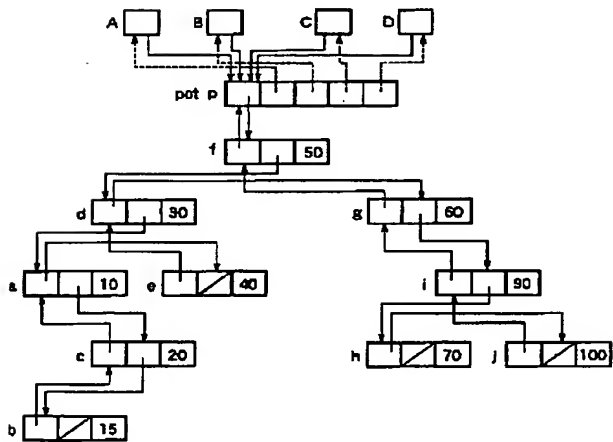
【図17】



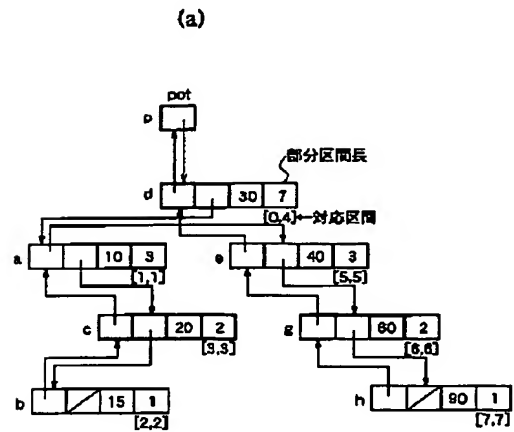
【図18】



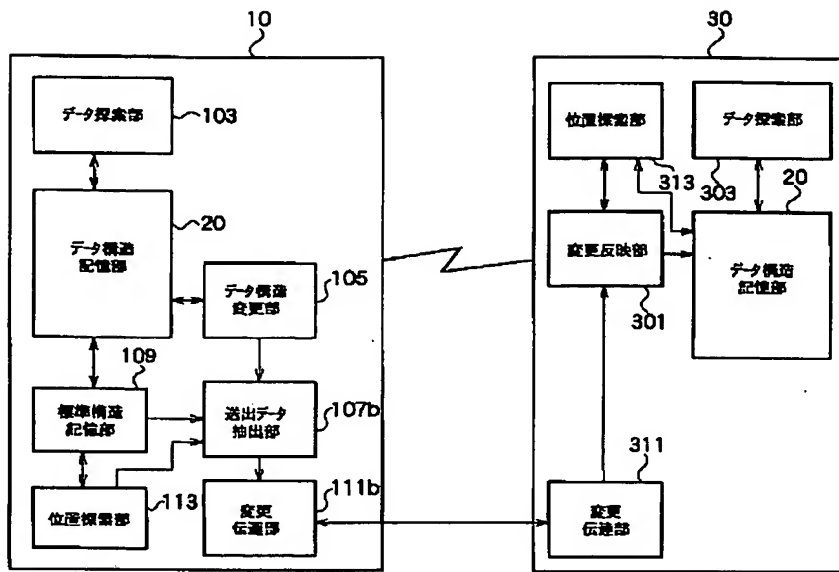
【図19】



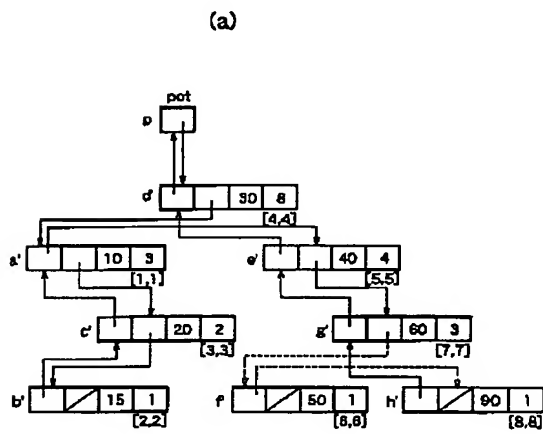
【図22】



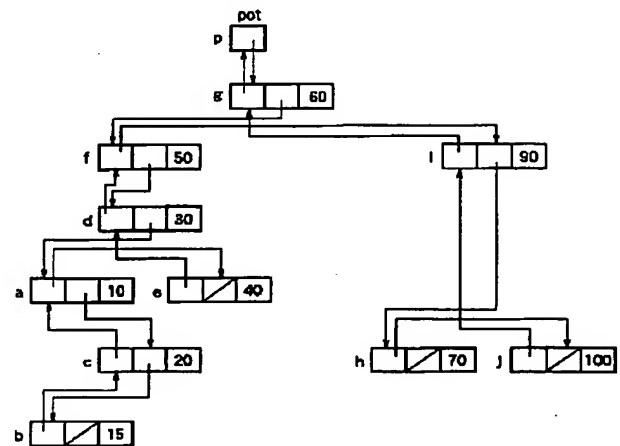
【図21】



【図23】



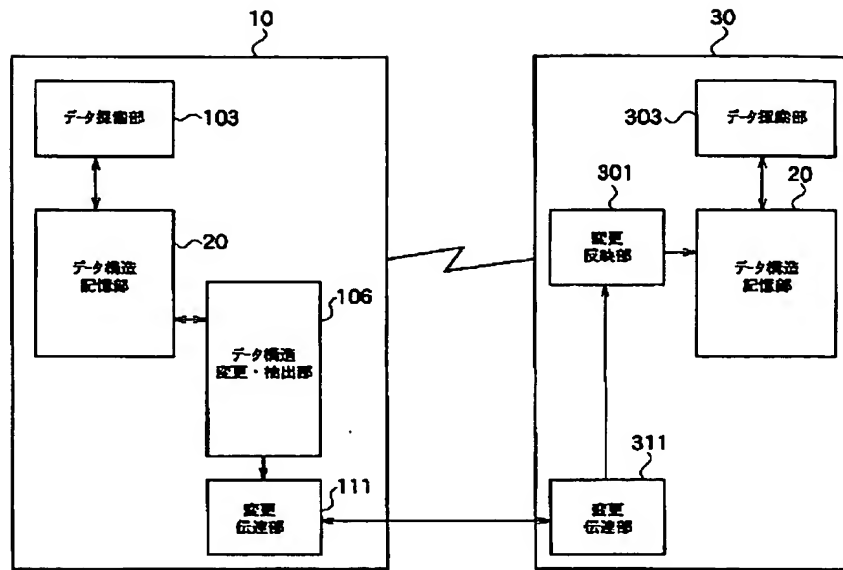
【図25】



(b)



【図24】



【図26】

